POCKET COMPUTER NEWSLETTER

© Copyright 1985 POCKET COMPUTER NEWSLETTER

Issue 40 - May/June

WORLD'S FIRST POCKET COMPUTER WITH TOUCH PAD SCREEN

Kiel Corporation, Inc., a one year old company based in Nashua, New Hampshire, has announced a hand-held computer that features a built-in combination display and touch pad.

The model, dubbed the Touch Pad I, features five display lines capable of displaying 25 characters each. Each display line also serves as a touch pad having 15 touch sensitive areas per line.

The unit also has a number of other unusual features. A removable Program Memory Card permits the Touch Pad I to be expanded to a maximum memory capacity of 120 kilobytes. Up to 56 kilobytes of this may be RAM elements, the remaining 64 kilobytes consisting of ROM. The Program Memory Card is available in 8K increments. The Program Memory Card is just 3 millimeters thick and about the size of a standard business card. Memory cards may be programmed using IBM PCs or similar microcomputers as well as by downloading to RAM via the unit's optional modem.

The Touch Pad I can be equipped with an *internal* modem capable of operating at either 1200 or 300 baud. Programs and data may be loaded into or transferred out of the unit via the modem.

Photo *Kiel Corporation's Touch Pad I Hand-Held Computer.*

The hand-held Touch Pad I measures just 6 by 4 by 3/4 inches. It accepts 7.5 to 12 volts direct current (unregulated) from an external source or it may be powered by a removeable battery pack. Current drain is 300 milliamperes (maximum) in the standard configuration. A lower power CMOS version is also available.

The unit utilizes an 8031/51 microprocessor. It comes standardly equipped with 2 kilobytes of RAM and 4 kilobytes of ROM. Kiel Corporation supplies a "program generator" that enables users to develop programs for the package on a larger system such as a desktop computer.

The liquid-crystal display can show variable width characters. Each line contains 960 pixels (120 by 8). Each line is also sectioned into 15 touch sensitive areas that may be used for input. Thus, the display can serve as a keyboard, keypad, function panel, etc., under the control of software.

The unit is also equipped with four user-defined function keys. And, it has a dual tone buzzer for providing audible alerts.

The Touch Pad I can also be optionally equipped with a bar code reader.

The manufacturer considers the product ideally suited for applications such as: building security and control, keyboard and interface to complex instrumentation (such as medical equipment), home banking terminal, weighing and packaging, factory data collection, service reporting, time management activities, robotics, point of sale terminal and a variety of control and monitoring tasks.

The single unit price of this unique touch-sensitive hand-held computer starts at $650.00. In quantity, such as in OEM applications, the price for a basic unit drops down to the $200 range.

Kiel Corporation is developing other products that take advantage of its proprietory touch-sensitive technology. Currently under development are liquid-crystal display touch panels that include "smart drivers" capable of displaying 40 characters per line. These panels can be equipped with 4 or 8 lines with all electronics integral to the panel. Such a panel, equipped with 8 lines of 40 characters, can contain 160 touch-sensitive positions. The company also expects to release a graphics panel that incorporates touch-sensitive features.

To obtain more information on the Touch Pad I, contact: **The Kiel Corporation,** *472 Amherst Street, P.O. Box 6340, Nashua, NH 03063.* The phone number is (603) 881-8666.

Another *Personal Information*™ product.

FOR PC-1250/51/60/61 USERS

MONITOR PROGRAM FOR 1250/1251

If you really like to dabble with using machine language in your Sharp PC-1250, this program can make life a lot easier than resorting to PEEKs and POKEs. The program was crafted by *Rick Wenger, 6221 - 18th Avenue, Kenosha, WI 53140,* who also supplies the following information about its use and operation.

Loading The Program

Load in the BASIC portion of the monitor from the accompanying listing. Enter the program *exactly* as shown. The program is address-dependent so do not attempt to add or delete anything.

After the BASIC part has been entered, switch to the RSV (reserve memory) mode and set up the following two reserve keys:

SHIFT C: CALL 50312
SHIFT N: CALL 50291 a

Now switch back to the PRO mode (or RUN mode) and use POKE directives to start loading in the machine code portion as shown in the accompanying listing. As indicated in the listing, once you get to a certain point, you can switch over and start using the monitor program itself to key in the balance of the machine code!

Of course, you can store the program on tape. Use a standard CSAVE command to save the BASIC part. Use the statement: CSAVE M "MLMON1";&C40F,&C5CF to store a copy of the machine code portion on tape.

Monitor Instructions

The monitor program provides the following capabilities by selecting the appropriate "defined" key.

DEF M: Preceded by a four-digit hexadecimal address puts you in monitor mode. Thereafter you may move to any address by keying the four digits (of the address) and hitting the ENTER key. Pressing ENTER without an input advances the current address by four. (You can change the number of locations-per-advance within the range 0 through 9 by keying a single digit followed by ENTER. An advance of five is useful when viewing the display memory. An advance of one is nice when entering machine code.) Entering machine language into RAM is accomplished by simply positioning the monitor to the desired address and entering a two-digit hexadecimal code followed by the ENTER key.

DEF SPC: Decrements the monitor address by the current locations-per-advance amount.

DEF F: Preceded by BRK, two digits, fills the temporary CPA with the specified hexadecimal byte. When not preceded by BRK and two digits, it just moves the monitor to the temporary registers area.

DEF B: Puts a breakpoint at the current monitor address.

DEF D: Deletes the last breakpoint reached.

DEF J: Preceded by BRK, four digits, causes a jump to the specified hexadecimal address. When not preceded by BRK and four digits, jumps to the previous DEF J address. Acts as a continue function when used immediately after DEF D. The command DEF J *must* be used if a breakpoint is present in a program. It should *not* be used if a routine ends with a RTS instruction.

DEF S: When preceded by BRK, two digits, searches for the specified hexadecimal byte starting from the current monitor address. If not preceded by BRK, two digits, resumes search for previously indicated search byte from the current monitor address.

DEF H: Provides hardcopy of memory contents via the printer. Output begins at current monitor address unless the command is preceded by BRK and four digits representing the desired starting address. Printing continues until user hits BRK or top of memory (FFFF) is reached.

DEF =: Converts the decimal number in the display to its four-digit hexadecimal equivalent.

SHIFT C: Pseudo-Clear function. Must be used in place of CLEAR in your BASIC programs that reside with the monitor program. Protects the machine language portion of the monitor. Should also be at the head of your program if you intend to use RUN or dimension any variables.

Program ***BASIC Portion of PC-1250/1251 Monitor.***

```
 1:"Y" CALL 50611:D=&00
   00: RETURN
 2:GOSUB "Y": POKE C,D:
   C=C+1
 3:""B= INT (C/256):
   POKE 50426,C-256B,B:
   CALL 50528
 4:B=0: INPUT "0F  0018
   :02 5C 32 4C ";B$:
   ON LEN B$ GOTO 11,2,
   3,6
 5:C=C+A: GOTO "
 6:"M" AREAD B$: GOSUB
   "Y":C=D: POKE 49285,
   194:A=4: GOTO "
 7:"J" AREAD B$: IF LEN
   B$=4 GOSUB "Y":B=
   INT (D/256): POKE 50
   357,B,D-256B
 8:CALL 50329: GOTO 10
 9:"F" AREAD B$: IF LEN
   B$=2 GOSUB "Y": POKE
   50283,D: CALL 50282
10:C=50428: GOTO "
11:A= VAL B$: GOTO 5
12:" "C=C-A: GOTO "
13:"B" CALL 50247: GOTO
   "
14:"D" CALL 50226: GOTO
   "
15:"=" AREAD D:B= INT (
   D/256): POKE 50426,D
   -256B,B: CALL 50394:
   PRINT B$: GOTO "
16:"S" AREAD B$: IF LEN
   B$=2 GOSUB "Y": POKE
   50214,D
17:CALL 50191:C= PEEK 5
   0426+256* PEEK 50427
   : GOTO "
18:"H" AREAD B$: IF LEN
   B$=4 GOSUB "Y":C=D
19:A=5: POKE 49285,159:
   GOTO "
```

Program *Machine Code Portion of PC-1250/1251 Monitor.*

```
Execute CLEAR <ENTER>
Execute DIM Z$(224)*2 <ENTER>
Check MEM=491 here
POKE in the following hexadecimal code (prefix each value with & sign):
C4DA: 84 10 C4 FA 1A 84 78 C5 A1 82 10 C6 93 1B 85 78
C4EA: C5 A1 82 11 91 1B 02 F5 11 90 52 23 11 95 52 37
C560: 78 C4 DA 88 00 03 10 C6 91 18 88 10 C0 8B 19 85
C570: 00 04 63 E0 05 2A 09 82 13 04 0A 88 35 2C 03 88
C580: 18 88 78 C5 A1 10 C0 87 82 1B 02 C0 87 DB 02 8F
C590: 86 DB 89 00 03 78 C5 A1 26 DA 26 06 50 41 2B 0A
C5A0: 37 59 78 C5 A8 DA 59 58 64 0F 74 40 67 4A 3A 03
C5B0: 74 07 37 00 04 84 02 40 1E 88 10 C6 91 18 87 50
C5C0: 63 00 29 04 51 51 51 51 10 C0 40 00 03 19 37 FA
At this point you can activate the Monitor to key in the rest of the
machine code. Key in: C40E DEF M 1 <ENTER>
Now use the Monitor to enter the following hexadecimal values. (You do
not need to prefix the values with the & sign when using the Monitor.)
C40F: 10 C4 FA 84 1A 00 00 04 88 55 85 63 40 2A 07 82
C41F: 13 04 0A 88 35 88 63 00 29 12 10 C4 FA 84 1B 37
Key in: C432 <ENTER> (to change address when using the Monitor)
C432: 10 C4 2F 82 00 02 18 10 00 00 82 19 10 C4 3A 1A
C442: 85 11 B5 1B 37 10 C4 FA 86 1A 84 02 B7 DB 02 C4
C452: DA 02 79 82 00 02 07 06 19 10 C4 3B 86 53 11 3A
C462: 87 53 10 C4 2F 82 19 37 02 00
C46C: 00 60 10 C4 FF 1F 37 02 1C 03 C2 82 10 C6 E3 1B
C47C: 02 E0 10 C0 31 52 02 FF 10 C2 1C 52 02 08 03 C4
C48C: 10 C6 FC 82 1B 23 00 AF 10 C5 D0 1F 37 80 00 5F
C49C: 10 C5 00 19 11 02 52 10 C4 FC 57 30 11 FD 57 31
C4AC: 11 FE 57 32 10 C5 02 57 79 00 00 10 C5 02 52 20
C4BC: 10 C4 FC 52 21 11 FD 52 22 11 FE 52 80 10 C5 00
C4CC: 53 11 02 57 11 01 00 5E 81 19 02 58 32 37
To save machine language portion on tape: CSAVE M "MLMON1";&C40F,&C5CF
```

SHIFT N: Pseudo-NEW function. Initializes your BASIC program but retains and protects the monitor program. Can also be used immediately after a hasty NEW or a bad crash to retrieve the monitor from the grave. (You may have to manually key: CALL 50291 ENTER, if the crash blanks the RSV memory.

USER'S MEMORY MAP

Here is how memory is utilized when the monitor is installed in the PC-1250:

C031 - C21C: BASIC portion of the monitor. This part is not relocatable.

C21D - C407: Part of machine language portion of the monitor. Not relocatable.

C4FC - C4FF: Temporary CPU registers (P, P' and SP) and fill byte.

C500 - C55F: Temporary CPA (00 - 5F).

C560 - C5CF: Additional part of machine language portion of the monitor.

C5D0 - C67F: Variables Z through E. Available to user for use as variables or as machine language scratchpad area.

C680 - C69F: Variables D through A. Used by monitor.

Putting The Monitor Through Its Paces

First, try something simple. Key in 49494 DEF =. You should see C156. Now try keying &8888 + &2345 DEF =. Do you see ABCD? You have been using the hexadecimal calculator!

Now try keying CL (the "clear" key) and DEF M. You will be looking at the "hidden" ROM. The first four bytes should be: 4E A0 02 01. Hit ENTER to see the next four bytes: 12 5F DB DF. The byte to the far left in the display is the preceding byte (at address 0003 in this instance). Key DEF SPC to get back to address 0. Now, try a different increment value, say 5. Key the number 5 then hit ENTER several times. Key DEF SPC the same number of times to get back to the starting point.

Here is a simple machine language program. It will double any number in the hexadecimal range 0 through 7F:

ADDRESS	MNEMONIC	MACHINE CODE
C300	LDA (P)	59
C301	ADD(P) A	44
C302	BREAK	79 C4 B7

To enter the program into memory using the monitor, just key: C300 ENTER, 59 ENTER, 44 ENTER, DEF B.

The following discussion assumes that you have an understanding of the machine language instructions being used. (This program is not designed to communicate directly with BASIC.) DEF J will be used to jump to the program. Break will used to get back to the monitor. Thus, we can use the monitor for input and output. A program set up in this fashion is fairly (?) crash-proof, in that you do not need to worry about upsetting registers SP, (01), (16) or (58) through (5B).

Refer to the User's Memory Map provided earlier. Note that the temporary CPU registers at C4FC, C4FD and C4FE are transferred to the real CPU registers P, P' and SP and that the temporary CPA is transferred to the real CPA when the monitor is placed in operation. After execution of the program the monitor enables the temporary CPA to reflect the results of your program (while guarding against a crash that could occur if the CPA was manipulated directly).

To verify operation of the example program and the monitor, key: BRK 13 DEF F. The monitor address should show C4FC. Key 03 ENTER so that P will point to register (03) in the CPA. Key in whatever you want for P' and SP (these will remain unchanged during the following operations). Verify that C4FF through C55F contain 13. Now key: BRK C300 DEF J.

If all is well, the program has now executed and you are back at C4FC. C4FC should still be 03 and C500 through C55F all still contain 13 -- except for C503 -- which contains 26. Key DEF J again and C502 will become 26 while C503 goes to 4C. You can try different values for P and (P). The only registers affected by this program are the accumulator (represented by C502) and (P) which is represented by C500 + (C4FC).

To confirm breakpoint operation, load (P) with 05. Position the monitor to C301. Key DEF B. Verify that C301 - 03 contains 79 C4 B7. Key DEF J. Verify that A and (P) contain 05. Key DEF D. See that C301 - 03 now contains 44 79 C4. Load A with 02. Key DEF J. Verify that (P) now holds 07. If you realize why this is so, you have a good understanding of monitor operations. (This discussion refers to CPA locations. You must, of course, examine the temporary register equivalents.)

You may explore the DEF S and DEF H operations at your leisure.

Installing The Monitor In The PC-1251

You need to make a few modifications to adapt the program as shown for the PC-1250 for the PC-1251. First, change the POKE value in lines 6 and 19 of the BASIC listing from 49285 to 47237. Next, in the machine code listing, change all the underlined values of C0 to B8 and the underlined values of C2 to BA. That's all it takes!

A Bare Bones Monitor

You can make a skeleton version of the monitor which lacks the functions DEF F, DEF SPC, DEF B, DEF D, DEF =, DEF $ and DEF H. But, you can still view memory and enter machine code. Why, you can even enter breakpoints by hand! It might come in handy in situations where memory is really at a premium. To make the "bare bones" version, do the following:

1. Delete lines 9 through 19 from the BASIC listing.
2. Change line 8 to read: CALL 50329:C=50428:GOTO "
3. In line 4, (after the ON GOTO), change the number 11 to a quotation mark (").
4. Change locations C46C - C471 to: EA 01 61 AE 00 02.
5. Change C474 to 21, C476 to C1, C486 to 21, C485 to C1 and C489 to 6C.

PC-1500 TO PC-1250 THROUGH PC-1261 PROGRAM CONVERSIONS

Quite a few readers have asked whether programs previously published in *PCN* for the model PC-1500 can be converted to run on a PC-1250/51/60/61. The fact is, many of the programs written in BASIC for the PC-1500 can also be run on the PC-1250 series with little change. The limiting factors are generally the amount of memory needed, whether or not plot capabilities are being utilized (as the equivalent functions are not available on the 1250 series), and the amount and type of variable usage. Let's take a look at each of these factors to try and gauge their effect on adapting a program from one type of machine to the other.

Memory: Remember, a PC-1500 can be expanded by an extra 8 to 16 kilobytes. Large programs written for the PC-1500 simply won't fit in a 1250 series PC that doesn't have sufficient memory. As a rule of thumb, allowing some margin for handling conversion problems, you can fit a 1500 byte PC-1500 program into a PC-1250, a 3000 byte PC-1500 program into a PC-1251 (or Radio Shack PC-3A) and a 9000 byte PC-1500 program into a PC-1261. These byte counts include storage used by any arrays being utilized in the program(s).

Printer/Plotter: The CE-150 printer/plotter interface typically used with the PC-1500 has capabilities that can not be duplicated on the CE-125 or CE-126P printer. Examine programs you are thinking of converting to see if they use plotter capabilities. If so, can the program still be useful if those capabilities are dropped? If not, look for another program to convert!

Variable Usage: A major difference between the PC-1500 and the 1250 series is how the primary variables (standard variables A - Z and string variables A$ - Z$) are allocated. In the PC-1500 these variables are completely independent. There are two sets (one for variables A - Z, the other for string variables A$ - Z$). Furthermore, the string set in the PC-1500 has room for 16 characters in each element. Such is not the case in the PC-1250 series. There is one set of memory locations that serves to hold the variables A - Z or A$ - Z$. If a variable is being used to hold a numerical value, it cannot simultaneously serve to hold a string value and vice versa. Furthermore, when a variable is assigned to serve as a string element, only 7 characters can be stored. This reduced capability on the PC-1250 series can make it difficult to convert a program that utilizes the standard string registers extensively in a PC-1500 application. However, such adaptations can be made by assigning two-character variables (such as A1$, B1$, C1$,... etc.) or array elements (such as A$(0), A$(1), A$(2),... etc.) in the PC-1250 series in place of the standard string variables (A$, B$, C$,... etc.) in the PC-1500.

Here are a few other tips you might want to consider when adapting programs from one model to the other: Watch out for "implied" multiplication. While the PC-1250 will interpret the statement AB as A times B, the PC-1260 would think you meant the two-character variable named AB. It is a good idea to always specifically invoke multiplication by using the multiplication (*) sign such as: A*B. Don't forget you can simply invoke printer operation on the PC-1250 series by giving the manual command PRINT = LPRINT. You can go back to using the display (for all print statements) by using the statement PRINT = PRINT. Thus, you do not have to complicate your PC-1250 programs with a whole bunch of flags and two sets of PRINT and LPRINT directives.

Simultaneous Equations

For those of you who like programming in BASIC (instead of machine language) here is a small but powerful program. It was originally developed by *Norlin Rober* for the Sharp PC-1211. But, as you shall see, it runs fine on a PC-1250, a 1251, 1260 or 1261. You use it to solve a system of equations having up to 12 unknowns. It only takes a little over 400 bytes for the program itself. Another 1300+ bytes are used by an array.

Use DEF/A to start the program. Respond to the prompts to input the number of equations and the coefficients for each one. For example, if you wanted to solve the system:

```
3W + 2X + 6Y + 2Z = 2
 W + 4X + 3Y + 2Z = 6
2W +  X + 9Y + 3Z = 2
4W + 3X + 6Y +  Z = 8
```

You would respond to the first prompt with 4 (for the number of equations) and then enter the coefficients in four sets as: (3, 2, 6, 2, 2), (1, 4, 3, 2, 6), (2, 1, 9, 3, 2) and then (4, 3, 6, 1, 8). If you have loaded the program properly you should obtain the solution: -2, 3, 2, -5.

If the coefficients entered are not acceptable, you will see the message REJECTED appear on the display. If so, enter any remaining sets of coefficients and then try re-entering the troublesome set. If the REJECTED message comes up again then the algorithm cannot solve the particular problem due to inconsistency or dependence. (The algorithm employs Gaussian elimination.)

Program *Simultaneous Equations*.

```
1:"A" DIM A(172):
  INPUT "NO. OF EQUATI
  ONS? ";A:A=A+1:B=0
2:"C" BEEP 1: FOR C=7
  TO A+6: INPUT A(C+A*
  B): NEXT C: IF B=0
  THEN 5
3:FOR D=1 TO B:E=A(A+A
  *B-D+6):F=A(A*D-D+6)
4:FOR C=7+A*B TO A+C-1
  :A(C)=A(C)-A(C+A*D-A
  -A*B)*E/F: NEXT C:
  NEXT D
5:IF A(C-B-1)=0 BEEP 1
  : PRINT "REJECTED":
  GOTO 2
6:B=B+1: IF A-B-1 GOTO
  2
7:FOR B=1-A TO -1:F=6-
  A*B:E=A(F)/A(F+B):A(
  F)=E: IF B+2 GOTO 9
8:FOR C=1 TO -B-1:F=A*
  C+6:A(F)=A(F)-A(F+B)
  *E: NEXT C: NEXT B
9:BEEP 1: FOR B=1 TO A
  -1:E=A(A*A-A*B+6):
  PRINT B;"  ";E: NEXT
  B: END
```

FOR HEWLETT-PACKARD HP-71B USERS

CURVE-FITTING PROGRAM

Engineers and scientists frequently need to fit a series of x,y data points to curves. What curve should you try? Linear? Logarithmic? Exponential or power? This powerful program can help you out!

This is a curve-fitting package that has been adapted specifically for the HP-71B. (The original version, designed for a Sharp PC-1500, was submitted by *Thomas S. Cox* and published in Issue 15 of ***PCM.***) Special thanks for assisting in the adaptation go to *Robert Findley*.

Load the program into your HP-71B from the accompanying listing. As a check on your accuracy, look for a byte count of 2329 upon keying in the program exactly as shown. Start the program using the RUN key.

Once running, you will see a six-item menu flashing on the display. Press the number key corresponding to the selection of your choice. Normally, you will start with item number 1 to enter a group of x,y points. You also make this selection if you want to add additional points to your curve later. (Select the CLEAR option if you are inputting a new set of data points. Respond negatively [N] to this prompt if you wish to add data points to previously entered data.)

To enter data points, respond appropriately to the prompts for point number (PT#), x- and y- values. Enter as many points as you want to obtain a decent fit. Input point number zero (PT# 0) to conclude the data entry operation. This will return you to the primary rotating menu.

Select menu item number 2 if you want to change data. You can eliminate or alter your latest entry point using this option. Just respond appropriately to the program's queries.

Menu item number 3 will automatically determine the best fit of your data to the four supported curves: linear, log, exponential or power. You will see the "fitting" procedure take place for each type of curve followed by the computer's selection of the best fit. The program then displays in order the constant equation term A, variable equation term B, coefficient of correlation R and coefficient of determination R^2 for the curve selected. (Press the END LINE key to obtain each term.) Once these have been presented

Program ***HP-71B Implementation of Curve-Fitting.***

```
10 DELAY .8,0 @ WIDTH 96 @ DISP "CURVE FIT MENU"
20 DISP "1-ENTER     2-CHANGE" @ IF KEYDOWN THEN 40
25 DISP "3-BEST      4-CURVE FIT" @ IF KEYDOWN THEN 40
30 DISP "5-EST X     6-EST Y"
40 IF NOT KEYDOWN THEN 20 ELSE Z$=KEY$
50 IF Z$<"1" OR Z$>"6" THEN 20 ELSE ON NUM(Z$)-48 GOSUB 80,480,380,60,420,550 @
60 DISP "CURVE MENU"
65 DISP "1-LINEAR   2-EXPON." @ IF KEYDOWN THEN 70 ELSE DISP "3-LOG       4-POWER
70 IF NOT KEYDOWN THEN 65 ELSE Z$=KEY$
75 IF Z$<"1" OR Z$>"4" THEN 60 ELSE ON NUM(Z$)-48 GOSUB 180,230,280,340
80 DISP "ENTER DATA" @ INPUT "CLEAR (Y/N)? ","N";Z$
90 IF Z$="Y" THEN DESTROY ALL
100 C=1 @ Z=N+1 @ INPUT "(0 TO STOP) PT# ";Z
105 Z$="X"&STR$(Z)&"= " @ IF Z=0 THEN 10 ELSE DISP Z$; @ INPUT "","?";L @ IF L<=
110 X=L @ Z$="Y"&STR$(Z)&"= " @ DISP Z$; @ INPUT "","?";Y @ IF Y<=0 THEN T=1
120 GOSUB 130 @ GOTO 100
130 D=D+C*X @ E=E+C*X*X @ F=F+C*Y @ G=G+C*Y*Y @ H=H+C*X*Y @ N=N+1*C @ IF J*T=1 T
140 IF J#1 THEN M=M+C*LOG(X) @ O=O+C*LOG(X)*LOG(X) @ K=K+C*Y*LOG(X)
150 IF T#1 THEN P=P+C*LOG(Y) @ Q=Q+C*LOG(Y)*LOG(Y) @ I=I+C*X*LOG(Y)
160 IF J+T=0 THEN S=S+C*LOG(X)*LOG(Y)
170 RETURN
180 IF W=1 THEN 210
190 U=1 @ DISP "LINEAR Y=B*X+A"
200 B=(H-D*F/N)/(E-D*D/N) @ A=F/N-B*D/N
210 R=(N*H-D*F)/SQR((N*E-D*D)*(N*G-F*F)) @ IF W=1 THEN RETURN
220 GOTO 600
230 IF W=1 THEN 260
240 U=2 @ DISP "EXP. Y=A*EXP(B*X)" @ IF T=1 THEN 430
250 B=(I-P*D/N)/(E-D*D/N) @ A=EXP(P/N-B*D/N)
260 R=(N*I-D*P)/SQR((N*E-D*D)*(N*Q-P*P)) @ IF W=1 THEN RETURN
270 GOTO 600
280 IF W=1 THEN 310
290 U=3 @ DISP "LOG. Y=B*LOG(X)+A" @ IF J=1 THEN 430
300 B=(K-M*F/N)/(O-M*M/N) @ A=(F-B*M)/N
310 R=(N*K-F*M)/SQR((N*O-M*M)*(N*G-F*F)) @ IF W=1 THEN RETURN
320 GOTO 600
330 IF W=1 THEN 360
340 U=4 @ DISP "PWR. Y=A*X^B" @ IF T+J>0 THEN 430
350 B=(S-M*P/N)/(O-M*M/N) @ A=EXP(P/N-(B*M/N))
360 R=(N*S-M*P)/SQR((N*O-M*M)*(N*Q-P*P)) @ IF W=1 THEN RETURN
370 GOTO 600
380 IF J+T>0 THEN 430
390 W=1 @ V=0 @ U=0 @ FOR Z=1 TO 4 @ ON Z GOSUB 180,230,280,330 @ DISP Z;R @ C=R
395 IF C>V THEN GOSUB 650
```

```
400 NEXT Z @ W=0 @ DISP "BEST FIT WAS #";U
410 ON U GOTO 180,230,280,330
420 DISP "ESTIMATE OF X" @ INPUT "Y= ","?";V @ ON U GOTO 440,450,460,470
430 DELAY 8,0 @ DISP "X, Y OR BOTH NEGATIVE" @ GOTO 10
440 L=(V-A)/B @ GOTO 630
450 L=(LOG(V)-LOG(A))/B @ GOTO 630
460 L=EXP((V-A)/B) @ GOTO 630
470 L=EXP((LOG(V)-LOG(A))/B) @ GOTO 630
480 INPUT "DEL. LAST PT (Y/N)?","N";Z$ @ IF Z$="Y" THEN 510
490 DISP "PT# ";N @ INPUT "X= ";L @ IF L<=0 THEN J=1
500 X=L @ INPUT "Y= ";Y @ GOTO 530
510 DISP "X=";X;"Y=";Y @ INPUT "OK TO DELETE (Y/N)? ","N";Z$
520 IF Z$="N" THEN 10
530 C=-1 @ IF Y<=0 THEN T=1
540 GOSUB 130 @ GOTO 10
550 DISP "ESTIMATE OF Y" @ INPUT "X= ";L @ ON U GOTO 560,570,580,590
560 V=B*L+A @ GOTO 630
570 V=A*EXP(B*L) @ GOTO 630
580 V=B*LOG(L)+A @ GOTO 630
590 V=A*L^B @ GOTO 630
600 DELAY 8,0 @ DISP "A="; @ DISP USING 700;A @ DISP "B=";
610 DISP USING 700;B @ DISP "R="; @ DISP USING 700;R @ C=R*R
620 DISP "R**2="; @ DISP USING 700;C @ GOTO 10
630 DELAY 8,0 @ DISP "X=";L;"Y=";V @ GOTO 10
640 DISP "X=";X;"Y=";Y @ RETURN
650 U=Z @ V=C @ RETURN
700 IMAGE 5D.DDD
```

the program returns to the main menu.

If you want to see how other curves fit, by examining equation terms and coefficient, select menu item number 4. This will bring up a secondary menu that enables you to select any of the four curves: linear, exponential, power or logarithmic. Select the curve desired. The terms and coefficients will then be presented in response to presses of the END LINE key.

Items 5 and 6 of the main menu permit you to obtain projected values of x and y based on the best fit. Respond to the prompts to obtain the desired information.

To test the program, try entering the following data points: PT# 1 X=1 Y=1, PT# 2 X=3 Y=3, PT# 3 X=5 Y=5. Remember, enter PT# 0 to end the data entering process. After returning to the main menu, select menu item number 3. Observe that the program displays four sets of data and then announces that "BEST FIT WAS # 1". It then displays the type of curve (Linear) and its equation. Finally, you should see that it reports the terms and coefficients as: A= 0.000, B= 1.000, R= 1.000 and R**R= 1.000. When back in the main menu, try items 5 and 6. Positive values for either axis (such as X= 9) should yield an identical figure for the other (i.e., Y= 9).

Note: when selecting from the flashing menu, hold the desired item number key down for about one-half a second.

The HP-71B is especially suited for scientists and engineers. This program can help people in those disciplines utilize some of its special capabilities.

FOR PC-1500 & PC-2 USERS

SPEED COMPARISONS -- PC-1500 STILL TOPS

Norlin Rober, 407 North 1st Avenue, Marshalltown, IA 50158, has run some comparisons amongst the Sharp and Radio Shack PCs. They show that the Sharp PC-1500, on the whole, outperforms all other compared models in terms of overall typical operating speeds. Here is his complete report.

The accompanying table shows the execution times (in milliseconds) for various types of operations performed on Sharp and Radio Shack models of pocket computers. To make the comparisons meaningful, the same input arguments for a particular operation were used on all computers.

Here is how the tests were conducted. First the simple line: 10 FOR X=1 TO 10000: NEXT X was executed. In the PC-1350 this took 69 seconds. Then, to check division for example, the statement A=B/C was inserted into that line. On the PC-1350, this was timed at 447 seconds. This indicated that

Table ***Comparisons of PC Execution Times.***

Statement Executed	PC-1211 PC-1	PC-1250 PC-3	PC-1260 PC-1261	PC-1350	PC-1500 PC-2
A=B	75	15	12.6	12.0	6.4
A=B+C	105	27	22.4	21.4	11.1
A=B*C	145	47	33.0	31.8	21.4
A=B/C	169	58	39.0	37.8	30.6
A=√B	171	67	43.4	42.5	39.8
A=B^C	630	408	252.3	257.6	205.9
Loop Cycle	192	42	7.5	6.9	14.0
GOTO (next line)	80	20	9.0	8.5	8.3
GOTO (Later line)	2420	161	26.1	25.6	16.0

NOTE: The "GOTO (Later line)" required searching past fifty 25-byte lines.

378 seconds was required for 10,000 division operations. Of course, division time will vary depending on what numbers are being divided.

Some observations that are of interest: 1. The newest computers such as the PC-1260 and PC-1350, surpassed the PC-1500 in executing loops, but they still *lagged behind the PC-1500 in all other tests!* 2. The PC-1350 outdid the PC-1260 by a small margin, except for one interesting case, when it handled the statement A=B^C slightly faster.

The "Benchmark" program listed in *PCW* Issue 26, page 4, was also tried. The results were as follows: PC-1211 = 693 seconds, PC-1250 = 218 seconds, PC-1260 = 119 seconds, PC-1350 = 118 seconds and the PC-1500 = 107 seconds. Again, the PC-1500 came in the overall winner.

In another comparison between the PC-1500 and PC-1350, using the same program in each PC to solve the same system of 8 linear equations, the PC-1500 took 22 seconds, the PC-1350 consumed 35 seconds.

Cassette operations, however, show a marked improvement in the PC-1260 and PC-1350. The accompanying table indicates the measurements made during tape operations (times are in seconds).

Table *Comparisons of PC Tape Operation Times.*

	PC-1211	PC-1250	PC-1260	PC-1350	PC-1500
Time for "Header"	6	8	8	8	11.7
Time, per kilobyte	93	44	27	27	72.5

A ROUTINE TO VERIFY MACHINE LANGUAGE SAVES

This program, CLOAD M?, is to machine language programming what CLOAD? is to BASIC -- a means of verifying that a program has been saved correctly on tape.

The program, supplied in machine language itself, is fully relocatable. Once you have keyed it into memory, be sure to save it on tape before using it.

To use the program, proceed as follows. First, save the machine language program you want to verify using the standard CLOAD M command. Rewind the tape to the start of that program. Place the remote switch on and press the "play" button on the recorder. Now, call the starting address of the CLOAD M? program. Respond to the TITLE: prompt with the name of the machine language program that you just saved. Press the ENTER key. The tape recorder should activate.

When the program has been read by the program, one of three displays will appear. VERIFICATION COMPLETE is shown if there were no discrepancies between the tape and the original code. DISCREPANCY will appear if there was any disagreement. In this case, you should try re-saving the code. The message CHECKSUM will appear if there was a read error. This indicates that the tape is bad. Re-record the code with a new tape.

This CLOAD M? program is supplied courtesy of: *Eric Bowman, PEA Box 81, Exeter, NH 03833.*

Program *CLOAD M? for the PC-1500/PC-1500A.*

```
7DBD F2 BE D0 2B
7DC1 FD 58 B5 08
7DC5 FD CA 6A 05
7DC9 8E 06 54 49
7DCD 54 4C 45 3A
7DD1 58 7B 5A B0
7DD5 F5 88 03 B5
7DD9 40 AE 78 80
7DDD BE E8 CA BE
7DE1 E2 43 C3 42
7DE5 B7 18 9B 2C
7DE9 EB 7B 0E 40
7DED B7 08 89 07
7DF1 5E B6 9B 15
7DF5 56 9E 1B B7
7DF9 0C 89 08 B5
7DFD 0D 17 9B 21
7E01 54 9E 27 B7
7E05 0D 8B 1D E9
7E09 7B 0E BF B7
7E0D 1C 89 05 BE
7E11 CD E6 9E 38
7E15 B7 1D 89 05
7E19 BE CE 38 9E
7E1D 41 B7 20 91
7E21 42 51 9E 48
7E25 B5 00 BE BB
7E29 D6 48 7B 4A
7E2D B6 58 7B 5A
7E31 69 05 B7 0D
7E35 8B 04 51 44
7E39 9E 09 B5 C0
7E3D AE 78 79 CD
7E41 B0 83 40 A5
7E45 7B AA 08 A5
7E49 7B AB 0A A5
7E4D 7B AC 28 A5
7E51 7B AD 2A CD
7E55 AA 83 2C CD
7E59 B4 F2 FD 58
7E5D B5 0A FD CA
7E61 FD 6A 4A 15
7E65 8E 15 56 45
7E69 52 49 46 49
7E6D 43 41 54 49
7E71 4F 4E 20 43
7E75 4F 4D 50 4C
7E79 45 54 45 BE
7E7D ED 3B BE D0
7E81 2B CD 46 87
7E85 23 F2 BE D0
7E89 2B CD B4 F2
7E8D FD 58 B5 0A
7E91 FD CA FD 6A
7E95 4A 0B 8E 0B
7E99 44 49 53 43
7E9D 52 45 50 41
7EA1 4E 43 59 BE
7EA5 ED 3B CD 46
7EA9 BE D0 2B CD
7EAD B4 F2 FD 58
7EB1 B5 0A FD CA
7EB5 FD 6A 4A 0E
7EB9 8E 0E 43 48
7EBD 45 43 4B 53
7EC1 55 4D 20 45
7EC5 52 52 4F 52
7EC9 BE ED 3B BE
7ECD E2 43 CD 46
```

FOR PC-1350 USERS

USING POKES ON THE PC-1350

The capability of the PC-1350 may be extended by the use of various POKES. A few of these techniques are discussed here.

Memory Reallocation

In an unexpanded PC-1350, BASIC normally starts at address &6030. This address is stored in a START OF BASIC pointer. This pointer is located at &6F01-02. Note that &6F01 contains &30, the low byte of this starting address. &6F02 contains &60, the high byte of this starting address.

BASIC can be relocated by changing the contents of this pointer. For example, execution of POKE &6F01,&30,&64, followed by execution of NEW (in the PRO mode) will set the START OF BASIC to &6430. Now any BASIC program, whether entered by CLOAD or from the keyboard, will be stored beginning at &6430. The area from &6030 to &642F is then available for machine language. BASIC will not interfere with it.

Execution of CALL 0 will restore the START OF BASIC pointer to its normal value, clearing programs and variables.

Those familiar with the PC-1500 and PC-2 will note that the two operations described above are equivalent to using NEW <address> and NEW 0.

Correction of the Printer Bug

If printing with the CE-126P is interrupted by the use of the BRK key, the computer does not automatically reset the

printer to the left margin. Thus, when printing is resumed, it may begin part way through the line. The left margin may be reset by execution of POKE &6F4E,0. (You might want to assign this POKE to a RESERVE key.)

Katakana Mode

The PC-1350 ROM contains codes for generating Katakana characters. Katakana mode is set by execution of POKE &6F16,&80. It is cancelled by switching the computer off then on. Or, by executing POKE &6F16,0. In Katakana mode, keying SML will display the Katakana annunciator rather than SML. The alphabet keys will then generate codes for Katakana characters. The SHIFT key prefix as well as the keys for plus, minus, equals, comma, colon and semicolon, allow keying in 63 Katakana characters. Note that the codes for each of these characters is a two-byte code, with &FE as the first byte. Hence, string variables will hold fewer than the normal number of characters of this kind. The Katakana alphabet is also obtainable using CHR$ 161 to 223. Also, CHR$ 241 to 244 will return some characters that apparently are not Katakana. Shift * will produce the symbol for Yen. All of these characters are printable with the CE-126P.

An Apostrophe Key

If you would like a key for the apostrophe, which may be displayed or printed, here is how it can be done. First, clear RESERVE memory by executing NEW in the RESERVE mode. Then POKE &7F7F,241,39. Now the key sequence SHIFT/SPC will key in the apostrophe character. (Additional RESERVE keys may be assigned in the normal way.)

How to TAB with the Printer

The PC-1350 BASIC lacks TAB. And, if you have tried it, you will have found that the CURSOR statement applies to the display, but not to the printer. Consider the printed line as consisting of 24 columns, numbered from 0 to 23. To begin the next following LPRINT in column n, execute POKE &6F4E,n. However, some precautions are in order. Do not use a value of n greater than 23 or you will have unexpected garbage printed. Also, do not set a column that precedes characters already specified for printing -- or some of the data will not get printed.

You might want to try this example: 10 LPRINT 1;:POKE &6F4E,7:LPRINT 2;:POKE &6F4E,15:LPRINT 3

Specification of Format

It is possible to specify PRINT (and LPRINT) formats with POKE rather than with USING. The advantage is that calculated values may be used to determine the desired format. The number of spaces to precede the decimal point, including a space for the sign, should be POKEd into &6F3A. The number of spaces to include and follow after the decimal pointer, should be POKEd into &6F3B. For character string lengths, POKE the number of spaces into &6F3C.

Keeping the Display Cursor in the Right Line

To set the display cursor to column n (with n ranging from 0 to 23) within the same line of the display that has already begun, execute CURSOR n, PEEK &788C. (The current line of the display being used by PRINT is stored in &788C.)

Some Fancy INPUT Prompts

When an INPUT statement follows a PRINTed prompting message, the INPUT will begin on the next line of the display. It is possible to accept input in the *same* display line as the PRINTed prompt by executing POKE &6F17,4. This enables the flashing cursor at the current location (rather than on the next line). A CURSOR statement is required after INPUT has been made, to advance to the next line. The following example should help clarify this procedure:

```
10 DIM A(10)
20 WAIT 0
30 FOR I=1 to 10: PRINT "A(";STR$ I;")=";
   POKE &6F17,4: INPUT A(I)
40 CURSOR: NEXT I
```

Clearing Defined Variables and Arrays

You can clear the defined variables and arrays *without* clearing the fixed variables by executing POKE &6F07,48,108.

Finally, if you forget your password, POKE &6F14,0 will reset the password protection feature.

The above information was supplied by: *Norlin Rober, 407 North 1st Avenue, Marshalltown, IA 50158.*

Available Only by Prepaid Subscription for a Calendar Year Period (January - December). You are sent back issues for the calendar year to which you subscribe, at the time you enroll.

__ Enroll me as a 1985 Subscriber (Issue numbers 37-44). $24.00 in U.S. (U.S. $30.00 to Canada/Mexico. Elsewhere U.S. $40.00 payable in U.S. funds against a U.S. bank.)
__ Enroll me as a 1984-85 Subscriber (Issue numbers 31-44). $42.00 in U.S. (U.S. $51.00 to Canada/Mexico. Elsewhere U.S. $70.00 payable in U.S. funds against a U.S. bank.)
__ Enroll me as a 1983-85 Subscriber (Issue numbers 21-44). $78.00 in U.S. (U.S. $93.00 to Canada/Mexico. Elsewhere U.S. $120.00 payable in U.S. funds against a U.S. bank.)
__ Enroll me as a 1982-85 Subscriber (Issue numbers 11-44). $102.00 in U.S. ($125.00 to Canada/Mexico. Elsewhere U.S. $160.00 payable in U.S. funds against a U.S. bank.)
__ Check here if paying by MasterCard or VISA. Please give credit card information below.

Orders must be accompanied by payment in full. All checks must be magnetically encoded, payable in U.S. funds and drawn against a U.S. bank.

Name: _____

Addr: _____

City: _____ State: ____ Zip: _____

MC/VISA #: _____

Signature: _____ Exp. Date: _____

mail this order form to:

POCKET COMPUTER NEWSLETTER
P.O. Box 232, Seymour, CT 06483

FROM THE WATCH POCKET

Is Sharp going to sell the PC-2500 in the United States? This is a portable "notebook" size version of the PC-1350. It has a touch-typing keyboard and a built-in printer/plotter along the lines of the CE-150 unit. The CPU is the same as that used in the 1350. It even uses the same memory expansion modules (CE-201 -- 8K and CE-202 -- 16K) as the PC-1350. There is a lot of potential interest in this unit, primarily because it might serve as a tool for PC software developers. Seems the machine is being delivered in Europe and other countries now, but is not yet available in the United States.

And, where is the technical manual for the PC-1350? We heard rumors months ago that Sharp would have a nice manual for the techies available soon.

If you are looking for PC-1500 supplies -- such as CE-161 (16K memory modules), printer/plotter paper & pens, etc., try contacting: **Atlantic N.E. Marketing,** *P.O. Box 921, Marblehead, MA 01945.* The phone number is: (617) 745-7707.